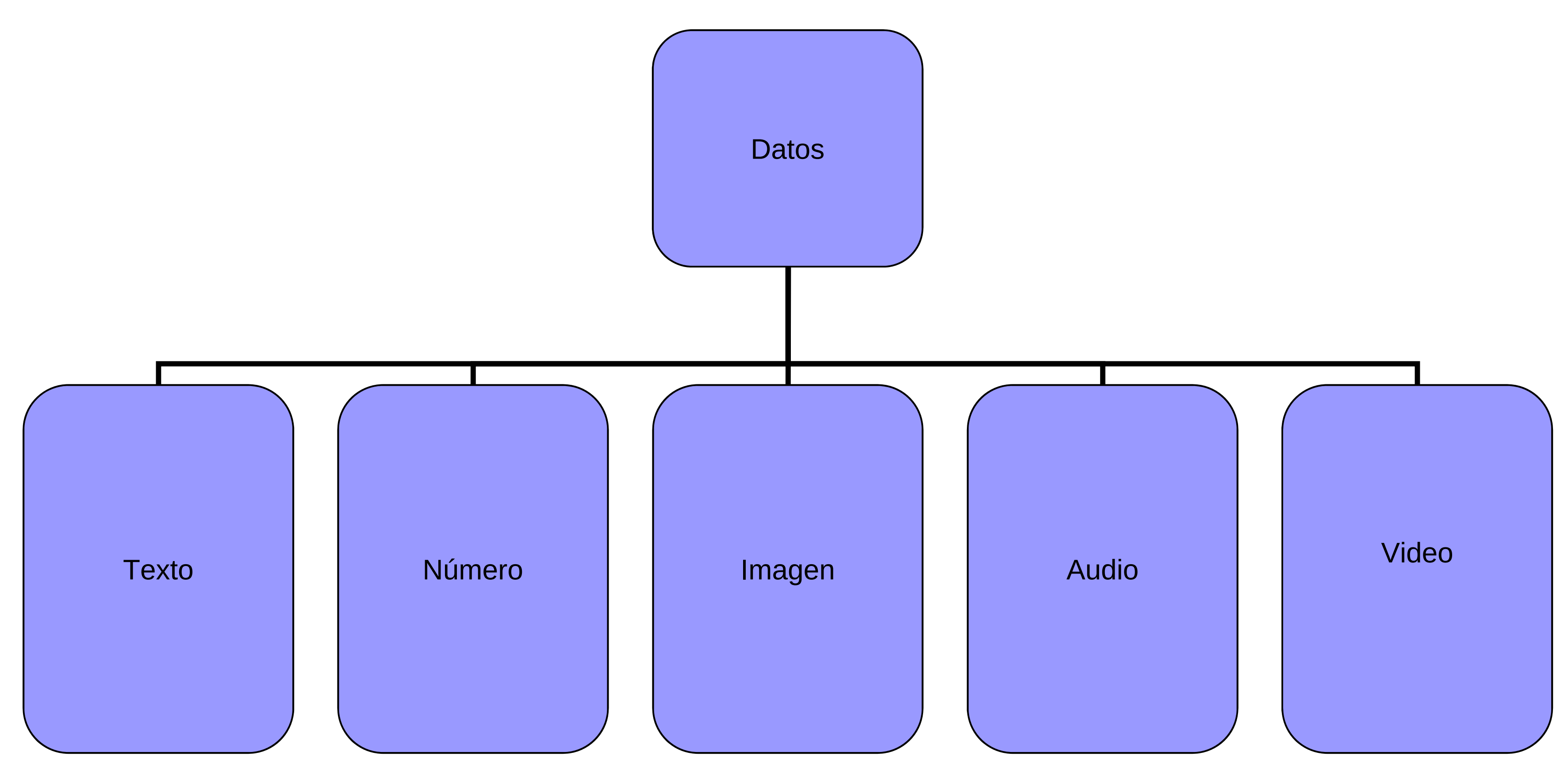
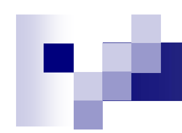
Representación de Datos

Una Introducción a los Sistemas Numéricos

Tipos de Datos



Multimedia: Información que contiene números, texto, imágenes, audio y video.



¿Como se manejan todos estos tipos de Datos?

- Los datos por lo general son una mezcla de tipos.
- Se utiliza una representación Uniforme de todos los tipos de datos.
- Los datos de entrada a una computadora se transforman en la representación uniforme al ser utilizados y almacenados por la PC.
- A esta representación uniforme o formato universal se le llama ***PATRÓN DE BITS.***

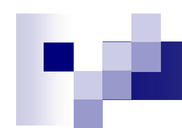
¿Qué es un Bit?

Binary Digit = Digito Binario

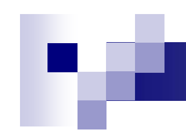
Unidad mas pequeña de datos que puede almacenarse en una computadora; puede ser ya sea 0 o 1 (cero o uno).

Un bit representa el estado de un dispositivo que puede tomar uno de dos estados. Por ejemplo un interruptor eléctrico.

Actualmente las computadoras utilizan varios dispositivos binarios de dos estados para almacenar datos.



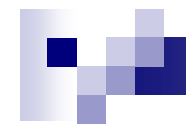
- Un solo bit no puede resolver el problema de la representación de datos, si cada pieza de datos (caracter) pudiera representarse por un 1 o un 0, entonces solo se necesitaría un bit. Sin embargo se hace necesario almacenar números mas grandes, texto, gráficos y otros tipos de datos.
- Es aquí donde se hacen necesarios los ***PATRONES DE BITS.***



Para representar diferentes tipos de datos se utiliza un PATRON DE BITS, una secuencia o como a veces se le llama, una cadena de bits. Ejemplo:

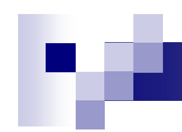
1 0 0 0 1 0 1 0 1 0 1 1 1 1 1 1

Esto significa que si se quiere almacenar un patrón de bits formado por 16 bits, necesitan 16 interruptores electrónicos. Si quiere almacenar 1000 patrones de bits, cada uno de 16 bits, necesita 16 000 bits y así sucesivamente.



¿Cómo identifica la memoria de la computadora el tipo de datos?

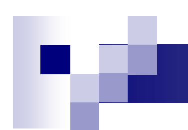
- No los identifica, la memoria solo almacena los datos como PATRONES DE BITS. Es responsabilidad de los dispositivos de entrada/salida o de los programas interpretar un patrón de bits como un numero, texto o algún otro tipo de datos.
- Los datos se codifican cuando entran a la computadora y se decodifican cuando se presentan al usuario.



¿Qué es un Byte?

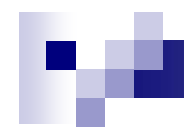
Un patrón de bits con una longitud de 8 se llama BYTE.

Este termino es también utilizado para medir el tamaño de la memoria o de otros dispositivos de almacenamiento.



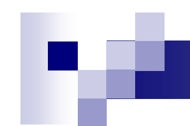
- Una pieza de texto en cualquier idioma es una secuencia de símbolos usados para representar una idea en ese idioma. (ej, ABC…Z, 0,1,2,3…9)
- Se puede representar cada símbolo (de lenguajes humanos) con un patrón de bits (lenguaje de la maquina)

B	Y	T	E	b	y	t	e
1000010	1011001	1010100	1000101	0110010	1001001	1000100	0110101



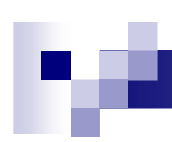
¿Cuántos bits se necesitan en un patrón de bits para representar un símbolo en un idioma?

- Esto depende de cuantos símbolos haya en la secuencia (idioma). La longitud del patrón de bits que representa un símbolo en un idioma depende del número de símbolos usados en ese idioma. Mas símbolos significan un patrón de bits mas grande.



¿Cuántos bits se necesitan en un patrón de bits para representar un símbolo en un idioma?

- Aunque la longitud del patrón de bits depende del numero de símbolos, la relación no es lineal, es logarítmica. Si se requieren dos símbolos, la longitud es un bit (el $\log_2 2$ es 1). Si se necesitan cuatro símbolos, la longitud es 2 bits ($\log_2 4$ es 2)



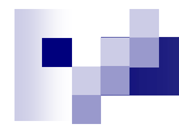
Numero de Símbolos	Longitud del Patrón de Bits
2	1
4	2
8	3
16	4
…	…
128	7
256	8
…	…
65 536	16

Un patrón de bits de dos bits puede tomar cuatro formas diferentes: 00, 01, 10 y 11

Cada una de las cuales representa un símbolo

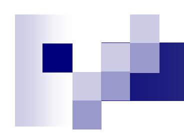
Un patrón de tres bits puede tomar ocho formas diferentes:

000, 001, 010, 011, 100, 101, 110 y 111



Códigos

Se han diseñado diferentes secuencias de patrones de bits para representar símbolos de texto. A esta secuencia se le conoce como Código y al proceso de representar los símbolos se le llama codificación

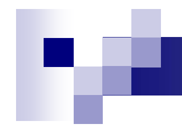


ASCII

Código Norteamericano de Estándares para Intercambio de Información *(American Standard Code for Information Interchange).*

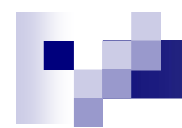
Este código utiliza siete bits para cada símbolo. Esto significa 128 (2^7) símbolos distintos que pueden definirse mediante este código

ASCII extendido: Para hacer que el tamaño de cada patrón sea de 1 byte (8 bits), a los patrones de bits ASCII se les aumenta un cero mas a la izquierda. Cada patrón cabe fácilmente en un byte de memoria



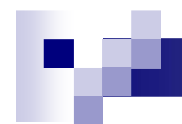
EBCDIC

- Código Extendido de Intercambio decimal codificado en binario desarrollado por IBM al principio de la era de las computadoras. Utiliza patrones de ocho bits, de manera que puede representar hasta 256 símbolos. Solo es utilizado en computadoras Mainframes de IBM.



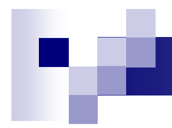
Unicode

- Ante la necesidad de un código de mayores capacidades, una coalición de fabricantes de hardware y software desarrollo un código que utiliza 16 bits y puede representar hasta 65 536 (2^{16}) símbolos. Diferentes seccione del código se asignan a los símbolos de distintos idiomas en el mundo.



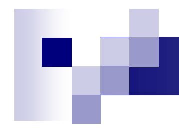
ISO

- La Organización Internacional para la Estandarización ha diseñado un código que utiliza patrones de 32 bits . Este código representa hasta 4 294 967 296 (2^{32}) símbolos, definitivamente lo suficiente para representar cualquier símbolo en el mundo actual.



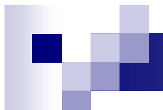
Números

En una computadora, los números se representan usando el sistema binario. En este sistema un patrón de bits (una secuencia de ceros y unos) representa un numero.



Imágenes

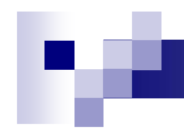
- Se representan en una computadora mediante uno de dos métodos: Gráficos de mapas de bits o gráficos de vectores



Imagen

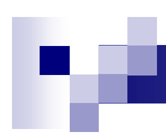
Bitmap

Vector

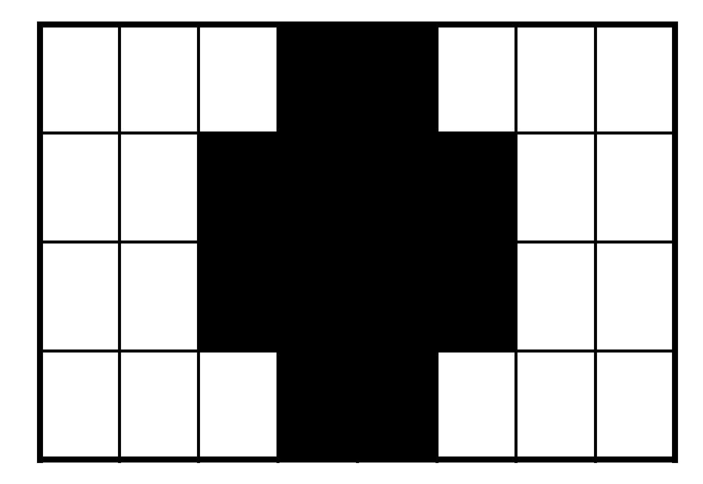


Gráficos de Mapa de Bits

Una imagen se divide en una matriz de pixeles. A cada pixel se le asigna un patrón de bits. El tamaño y el valor del patrón depende de la imagen, para una imagen formada solo por puntos blancos y negros, un patrón de un bit es suficiente para representar un pixel. Los patrones se registran uno tras otra y se almacenan en la computadora



Imagen

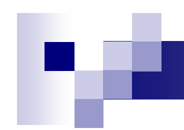


Representación de Pixeles

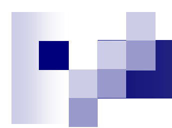
0	0	0	1	1	0	0	0
0	0	1	1	1	1	0	0
0	0	1	1	1	1	0	0
0	0	0	1	1	0	0	0

Representación Lineal

00011000 00111100 00111100 00011000

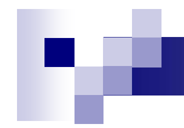


Para representar imágenes a color, cada pixel coloraedo se descompone en tres colores primarios: rojo, verde, azul (RGB). Luego se mide la intensidad de cada color y se le asigna un patrón de bits. En otras palabras, cada pixel tiene tres patrones de bits: uno para representar la intensidad del color rojo, uno para la intensidad del color verde y uno para la intensidad del color azul .



Representación de pixeles de color

Rojo	(con intensidad de 100%)	→	11111111 00000000 00000000
Verde	(con intensidad de 100%)	→	00000000 11111111 00000000
Azul	(con intensidad de 100%)	→	00000000 00000000 11111111
Blanco	(con intensidad de 100%)	→	11111111 11111111 11111111

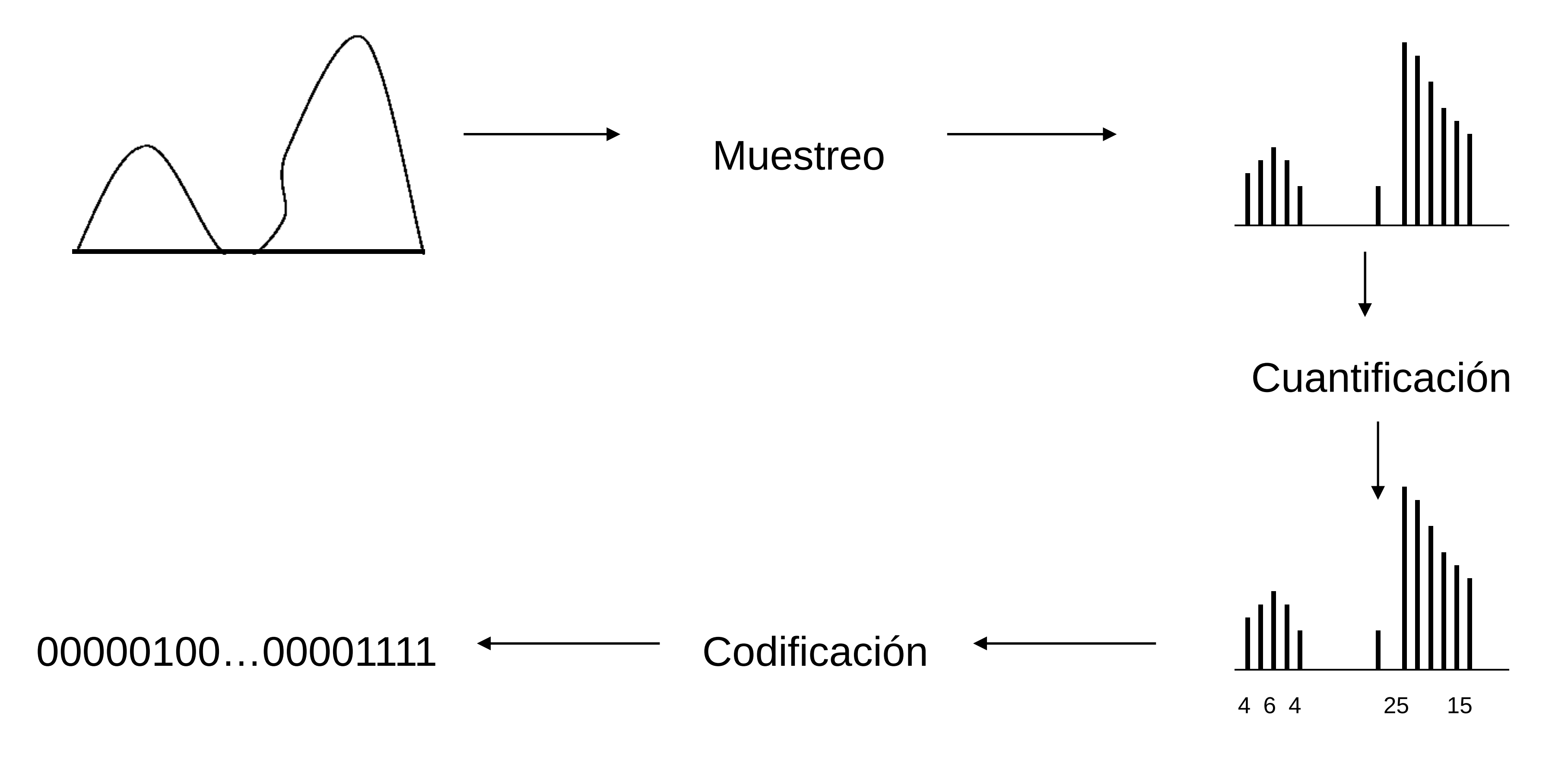


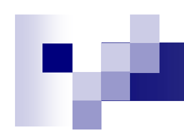
Representación de Gráficos de Vectores

Este método no guarda los patrones de bits. La imagen se descompone en una combinación de curvas y líneas. Cada curva o línea se representa por medio de una formula matemática. En este caso cada vez que se dibuja la imagen, la formula se vuelve a evaluar.

Audio

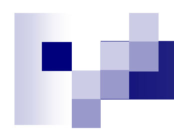
El audio por naturaleza es información análoga, es continuo no discreto.





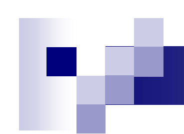
Video

Es una representación de imágenes (llamadas cuadros o frames) en el tiempo. Una película es una serie de cuadros desplegados uno tras otro para crear la ilusión de movimiento. Cada imagen o cuadro cambia a una serie de patrones de bits y se almacena. La combinación de las imágenes representa el video.



Notación Hexadecimal

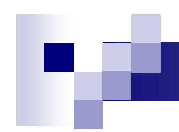
El patrón de bits se diseñó para representar datos cuando estos se almacenan dentro de una computadora. Sin embargo, para la gente es difícil manipular los patrones de bits. Escribir una series de números 0 y 1 es tedioso y propenso al error.



- La Notacion hexadecimal se basa en 16. Esto significa que hay 16 símbolos (dígitos hexadecimales): 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
- La importancia de la notación hexadecimal se hace evidente cuando se convierte un patrón de bits a esta notación.
- Cada digito hexadecimal puede representar cuatro bits y cuatro bits pueden representarse mediante un digito hexadecimal.



Dígito Hexadecimal	Patrón de bits	Dígito Hexadecimal	Patrón de bits
0	0000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111



Notación Octal

Otra notación usada para agrupar patrones de bits es la notación octal. Se basa en 8, esto es, existen ocho símbolos (dígitos octales): 0,1,2,3,4,5,6,7.

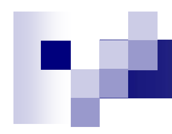
Cada dígito Octal representa tres bits y tres bits pueden representarse mediante un digito octal.



Dígito Octal	Patrón de Bits	Dígito Octal	Patrón de Bits
0	000	4	100
1	001	5	101
2	010	6	110
3	011	7	111

Sistemas Numéricos

Principios Básicos



Sistemas Numéricos

Un sistema numérico es un conjunto de números que se relacionan para expresar la relación existente entre la cantidad y la unidad. Debido a que un número es un símbolo, podemos encontrar diferentes representaciones para expresar una cantidad.

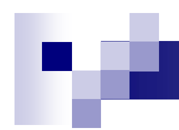
Se clasifican en posicionales y no posicionales.

Sistemas Posicionales

- Cada cifra de un valor numérico contribuye al valor final dependiendo de su valor y de la posición que ocupa. En estos sistema tenemos tantos símbolos como la base del sistema. Los números mayores que la base se representan por medio de varias cifras. El valor final será la suma de una serie de potencias de la base del sistema:

$$N = A_n * B^n + A_{n-1} * B^{n-1} + \dots + A_1 * B^1 + A_0 * B^0$$

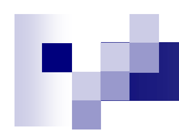
Donde A_i son las distintas cifras del valor numérico e “i” su posición.



Sistemas No Posicionales

Al contrario que en los sistemas posicionales, la contribución de cada cifra no depende del lugar que ocupa. Ejemplo, los números romanos.

El inconveniente que tienen estos sistemas es que para escribir valores numéricos grandes son necesarios muchos símbolos, y además resulta difícil efectuar operaciones aritméticas con ellos, cosa que no sucede con los posicionales.



Sistema Decimal

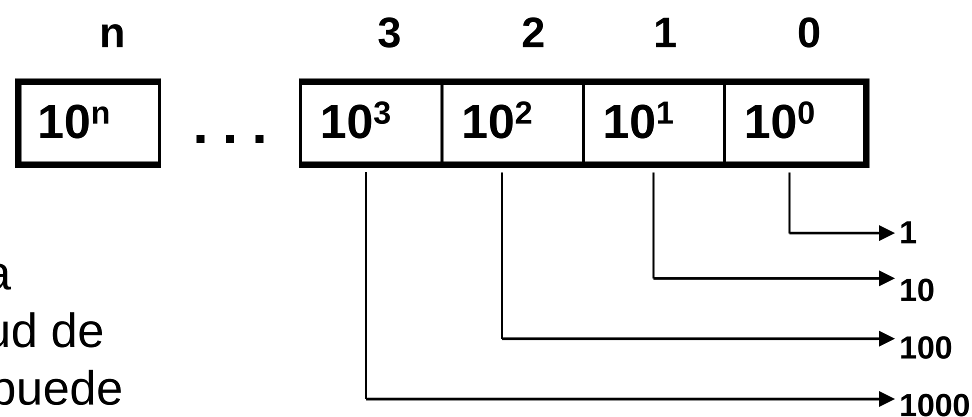
En el sistema de numeración decimal se utilizan diez símbolos, del 0 al 9 para representar una determinada cantidad. Los diez símbolos no se limitan a representar solamente diez cantidades diferentes, ya que se utilizan varios dígitos en las posiciones adecuada dentro de un numero para indicar la magnitud de la cantidad.

Base: 10

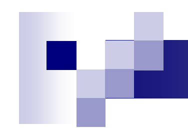
Símbolos: 0,1,2,3,4,5,6,7,8,9

Siguiendo la formula:

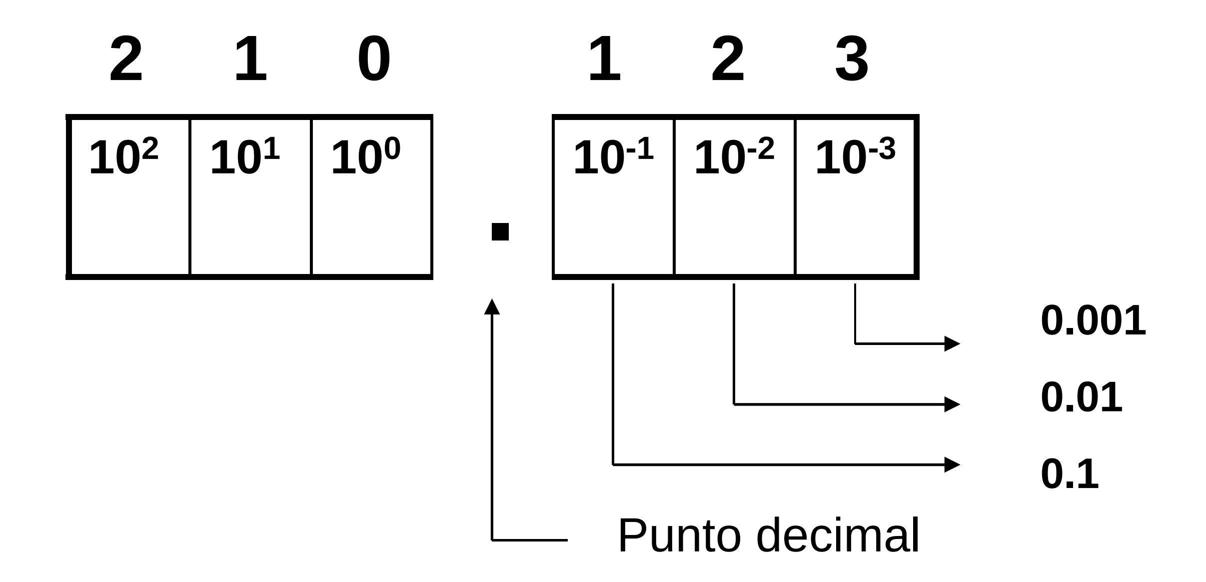
$N = A_n * B^n + A_{n-1} * B^{n-1} + \ldots + A_1 * B^1 + A_0 * B^0$

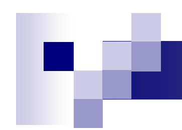


La posición de cada digito en una numero decimal indica la magnitud de la cantidad representada y se le puede asignar un peso. Los pesos para los números enteros son potencias de 10, que aumentan de derecha a izquierda, comenzando por $10^0 = 1$.



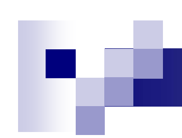
Para números fraccionarios, los pesos son potencias negativas de diez que aumentan de izquierda a derecha comenzando por 10^{-1}





Sistema Binario

El sistema de numeración binario es simplemente otra forma de representar magnitudes. Es menos complicado que el sistema decimal ya que solo tiene dos dígitos. Es un sistema en base dos. Los dos dígitos binarios son 0 y 1. La posición de un 1 o un 0 en un numero binario indica su peso dentro del numero, así como la posición de un digito decimal determina el valor de ese digito. Los pesos de un numero binario están basados en las potencias de dos.



Base: 2

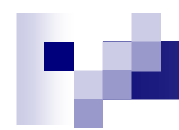
Símbolos: 0,1

El formato para números enteros y fraccionarios es similar al de los números decimales.

El sistema binario presenta el inconveniente de que necesita muchas cifras para la representación de un numero grande, y es muy engorroso para un humano.

Sin embargo, el sistema binario es el mas adecuado a las maquinas electrónicas por varias razones:

1. La mayor parte de las computadoras existentes representan la información y la procesan mediante elementos y circuitos electrónicos de dos estados (relés, núcleos de ferrita, etc.):
2. Por la seguridad y la rapidez de respuesta de los elementos físicos de dos estados diferenciados (ON / OFF).
3. Las operaciones aritméticas son sencillas.



Sistema Hexadecimal

Este sistema da una forma mas compacta para representar los números binarios. Consta de 16 símbolos. Para indicar que el número se expresa en hexadecimal se suela colocar una H al final, $34AF_{16}$ $34AF_{H}$

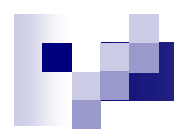
Base: 16

Símbolos: 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

Sistema Hexadecimal

Ejemplo de su uso en informática (de nominación de los colores para la web)

#000000	#000033	#000066	#000099	#0000CC	#0000FF	#0033FF	#0033CC	#003399	#003366	#003333	#003300
#006600	#006633	#006666	#006699	#0066CC	#0066FF	#0099FF	#0099CC	#009999	#009966	#009933	#009900
#00CC00	#00CC33	#00CC66	#00CC99	#00CCCC	#00CCFF	#00FFFF	#00FFCC	#00FF99	#00FF66	#00FF33	#00FF00
#330000	#330033	#330066	#330099	#3300CC	#3300FF	#3333FF	#3333CC	#333399	#333366	#333333	#333300
#336600	#336633	#336666	#336699	#3366CC	#3366FF	#3399FF	#3399CC	#339999	#339966	#339933	#339900
#33CC00	#33CC33	#33CC66	#33CC99	#33CCCC	#33CCFF	#33FFFF	#33FFCC	#33FF99	#33FF66	#33FF33	#33FF00
#660000	#660033	#660066	#660099	#6600CC	#6600FF	#6633FF	#6633CC	#663399	#663366	#663333	#663300
#666600	#666633	#666666	#666699	#6666CC	#6666FF	#6699FF	#6699CC	#669999	#669966	#669933	#669900
#66CC00	#66CC33	#66CC66	#66CC99	#66CCCC	#66CCFF	#66FFFF	#66FFCC	#66FF99	#66FF66	#66FF33	#66FF00
#990000	#990033	#990066	#990099	#9900CC	#9900FF	#9933FF	#9933CC	#993399	#993366	#993333	#993300
#996600	#996633	#996666	#996699	#9966CC	#9966FF	#9999FF	#9999CC	#999999	#999966	#999933	#999900
#99CC00	#99CC33	#99CC66	#99CC99	#99CCCC	#99CCFF	#99FFFF	#99FFCC	#99FF99	#99FF66	#99FF33	#99FF00
#CC0000	#CC0033	#CC0066	#CC0099	#CC00CC	#CC00FF	#CC33FF	#CC33CC	#CC3399	#CC3366	#CC3333	#CC3300
#CC6600	#CC6633	#CC6666	#CC6699	#CC66CC	#CC66FF	#CC99FF	#CC99CC	#CC9999	#CC9966	#CC9933	#CC9900
#CCCC00	#CCCC33	#CCCC66	#CCCC99	#CCCCCC	#CCCCFF	#CCFFFF	#CCFFCC	#CCFF99	#CCFF66	#CCFF33	#CCFF00
#FF0000	#FF0033	#FF0066	#FF0099	#FF00CC	#FF00FF	#FF33FF	#FF33CC	#FF3399	#FF3366	#FF3333	#FF3300
#FF6600	#FF6633	#FF6666	#FF6699	#FF66CC	#FF66FF	#FF99FF	#FF99CC	#FF9999	#FF9966	#FF9933	#FF9900
#FFCC00	#FFCC33	#FFCC66	#FFCC99	#FFCCCC	#FFCCFF	#FFFFFF	#FFFFCC	#FFFF99	#FFFF66	#FFFF33	#FFFF00



Sistema Octal

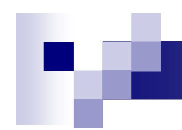
Este sistema tiene una base de ocho símbolos. La facilidad que existe en convertir entre el sistema binario y el octal, permite expresar los números binario en un formato mas compacto, ya que cada digito octal equivale a tres dígitos binarios.

Base 8:

Símbolos: 0,1,2,3,4,5,6,7

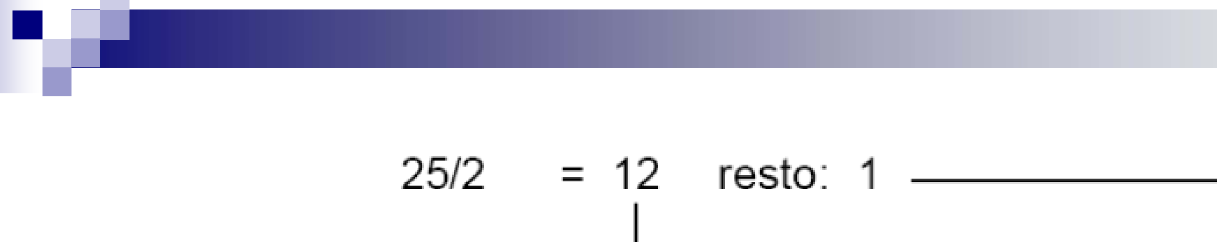
Sistemas Numéricos

Conversiones



Decimal a Binario

Se divide sucesivamente el numero decimal entre 2. Cada cociente resultando se divide entre 2 hasta que se obtiene un cociente cuya parte entera es 0. Los restos generados en cada división forman el numero binario. El primer resto es el bit menos significativo (LSB) del numero binario, y el ultimo resto es el bit mas significativo (MSB).



25/2 = 12 resto: 1

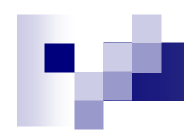
12/2 = 6 resto: 0

6/2 = 3 resto: 0

3/2 = 1 resto: 1

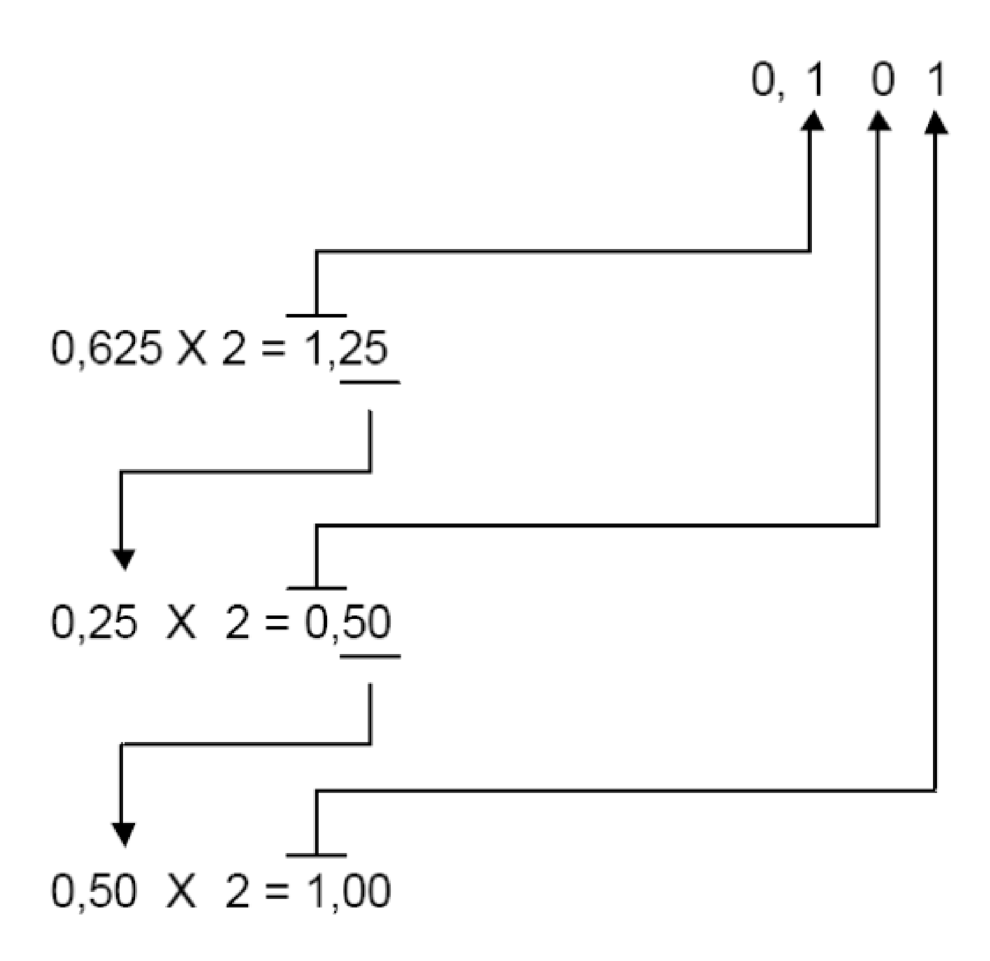
1/2 = 0 resto: 1

Resultado $25_{10} = 1\ 0\ 0\ 1\ 1_2$



Si el numero tiene **parte fraccionaria**. Esta parte se convierte utilizando la multiplicación sucesiva por dos, la cual consiste en multiplicar la parte fraccionaria por 2 y después se multiplica cada parte fraccional resultante del producto por 2, hasta que el producto fraccionario sea 0 o hasta que se alcance el numero deseado de posiciones decimales. Los dígitos acarreados, o acarreos generados por la multiplicación dan lugar al numero binario. El primer acarreo que se obtiene es el MSB y el ultimo el LSB.

Método de la multiplicación sucesiva:



El resultado final es la unión de ambos valores:

$109,625_{10} = 1101101,101_2$.

Binario a Decimal

La forma mas sencilla para realizar esta conversión es desarrollando la formula que vimos para los sistemas posicionales (suma de potencias de la base).

$$N = A_n * B^n + A_{n-1} * B^{n-1} + \ldots + A_1 * B^1 + A_0 * B^0$$

Donde A_i son las distintas cifras del valor numérico e "i" su posición.

B = 2

Ejemplo 1:

Dado el número binario: "1011_2", encontrar el equivalente decimal.

Si desarrollamos el número dado como potencias de 2 tendremos:

$$1011_2 = 1\cdot2^3 + 0\cdot2^2 + 1\cdot2^1 + 1\cdot2^0 = 1\cdot8 + 0\cdot4 + 1\cdot2 + 1\cdot1 = 8 + 2 + 1 = 11_{10}$$

Ejemplo 2:

Ahora vamos a realizar lo mismo pero con cifras decimales.

Dado el número binario: "$1011,011_2$", encontrar el equivalente decimal.

$1011,011_2 = 1·2^3+ 0·2^2+ 1·2^1 + 1·2^0+ 0·2^{-1}+ 1·2^{-2} + 1·2^{-3} = 1·8+ 0·4+1·2+1·1+0·1/2 + 1·¼ + 1·1/8 = 11,375_{10}$

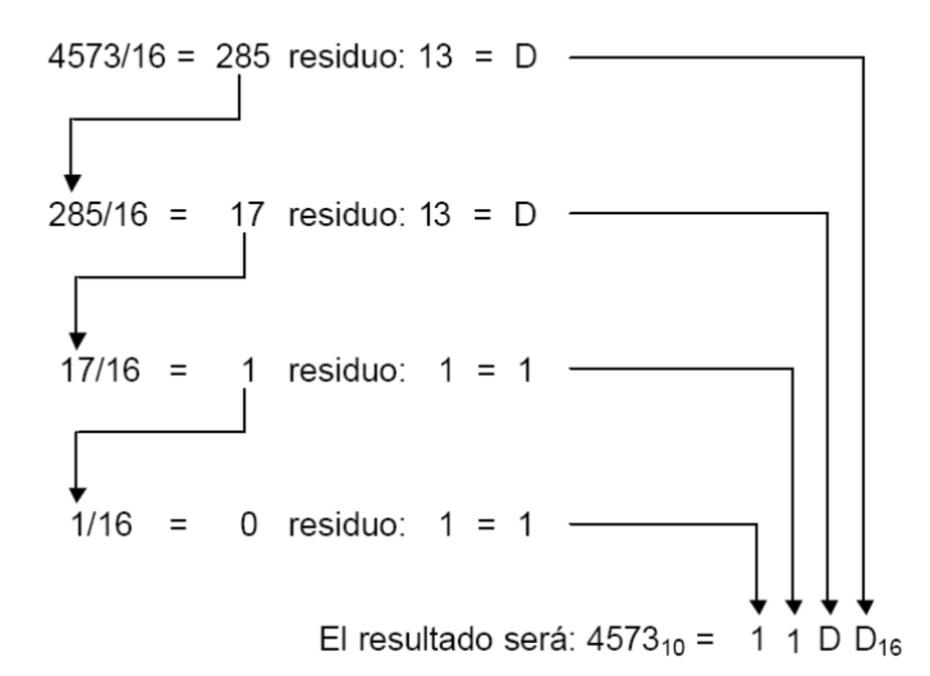
Decimal a Hexadecimal

Procederemos del mismo modo que en la conversión decimal-binario, considerando B=16. Dividiremos la parte entera sucesivamente por la base, y la parte fraccionaria la multiplicaremos por la base.

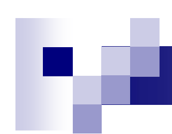
Ejemplo:

Hállese el equivalente hexadecimal del número $4573{,}79_{10}$.

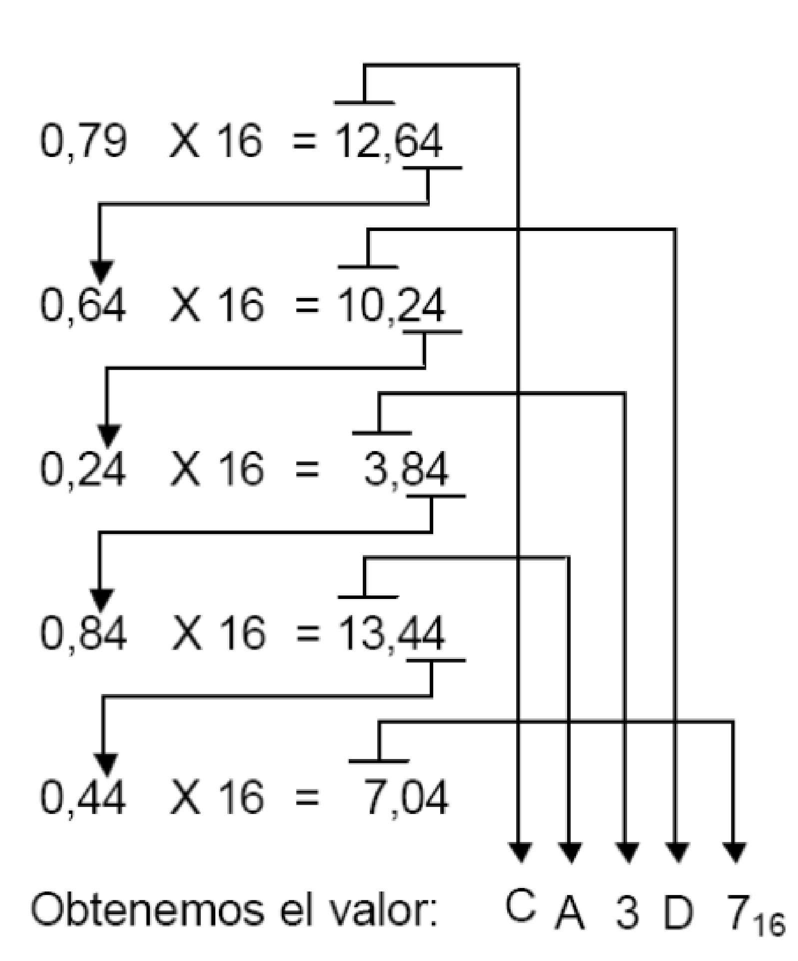
Parte entera



El resultado será: 4573_{10} = 1 1 D D_{16}



Parte fraccionaria



El resultado final es la unión de ambos valores:

$11DD,CA3D7..._{16}$

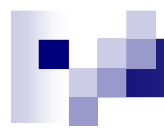
Hexadecimal a Decimal

La conversión se realiza siguiendo el mismo procedimiento que en las conversiones binario-decimal, pero considerando la base B=16. En este caso, además, deberemos sustituir los valores A, B, C, D, E, F por su equivalencia en el sistema decimal.

Ejemplo:

Hállese el equivalente decimal del valor hexadecimal $39{,}B8_{16}$.

$$\begin{aligned} 39{,}B8_{16} &= 3\cdot 16^{1} + 9\cdot 16^{0} + B\cdot 16^{-1} + 8\cdot 16^{-2} = \\ &= 3\cdot 16^{1} + 9\cdot 16^{0} + 11\cdot 16^{-1} + 8\cdot 16^{-2} = \\ &= 48 + 9 + 0.6875 + 0.03125 = \\ &= 57{,}71875 \end{aligned}$$



Hexadecimal a Binario

CONVERSIÓN HEXADECIMAL-BINARIO

Basta con sustituir cada símbolo hexadecimal por su equivalente en binario, según se indica en la tabla siguiente:

Hexadecimal	Decimal	Binario
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

Binario a Hexadecimal

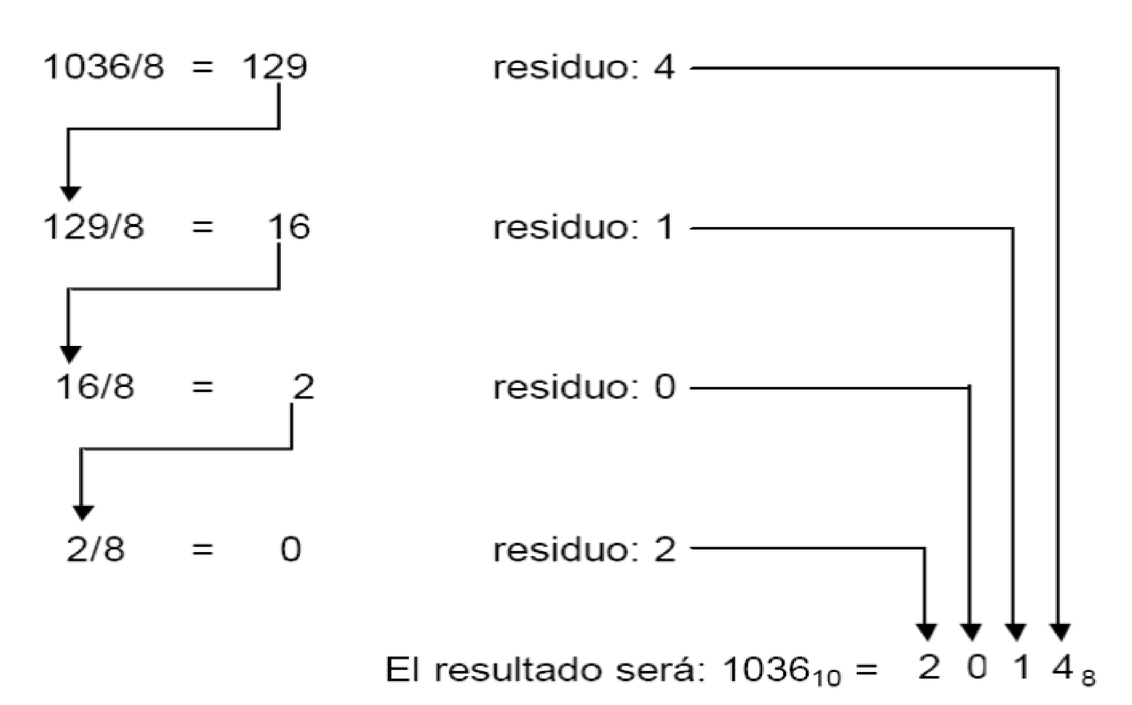
CONVERSION BINARIO-HEXADECIMAL

Basta con sustituir cada símbolo Binario por su equivalente en hexadecimal, según se indica en la tabla.

Hexadecimal	Decimal	Binario
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

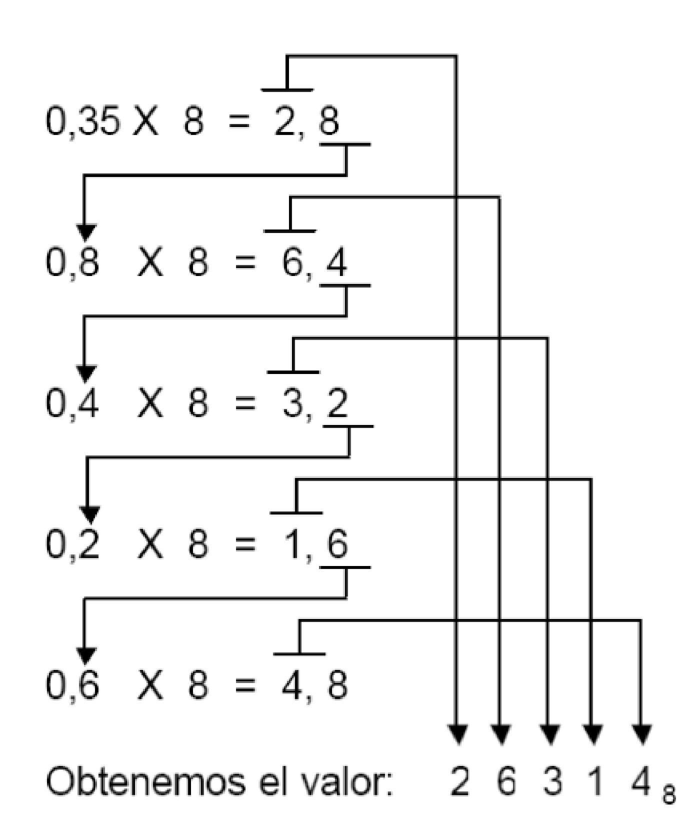
Decimal - Octal

Se realiza del mismo modo que de Decimal a Binario, dividiendo la parte entera de forma sucesiva por la Base = 8 , y multiplicando la parte fraccionaria por la base



Decimal - Octal

Parte fraccionaria



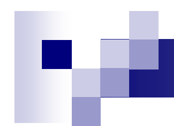
El resultado final es la unión de ambos valores:

$1036{,}35_{10} = 2014{,}26314\ldots_8$.

Octal - Decimal

Se realiza del mismo modo que de Binario a Decimal, teniendo en cuenta que la Base ahora es 8

Ejemplo: $345{,}5_8 = 3 \cdot 8^{2} + 4 \cdot 8^{1} + 5 \cdot 8^{0} + 5 \cdot 8^{-1} = 192 + 32 + 5 + 0{,}625 = 229{,}625_{10}$



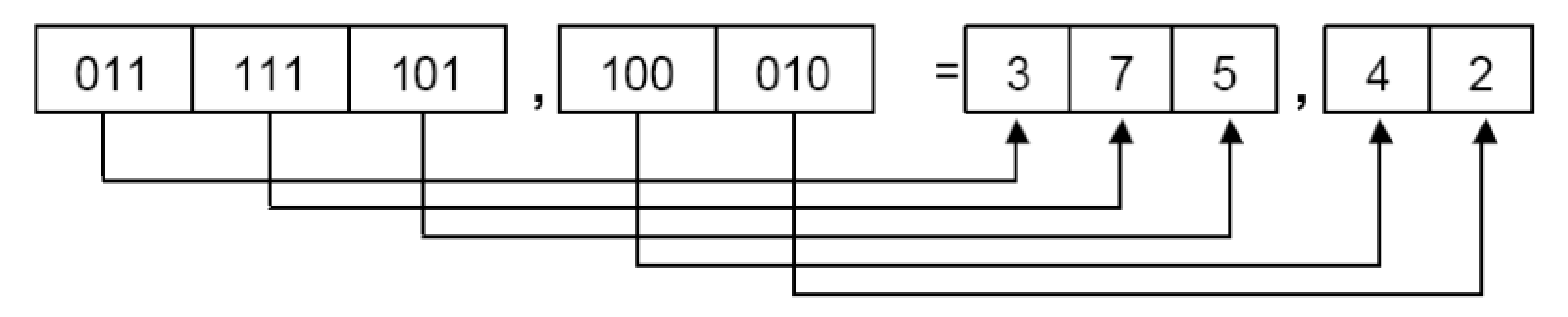
Equivalencias en el Sistema Octal

Octal	Decimal	Binario
0	0	000
1	1	001
2	2	010
3	3	011
4	4	100
5	5	101
6	6	110
7	7	111

Binario - Octal

Se realiza a la inversa, comenzando desde la coma decimal hacia la izquierda para la parte entera, rellenando con 0´s a la izquierda si fuera necesario; y desde la coma decimal hacia la derecha para la parte fraccionaria, rellenando con 0´s a la derecha si fuera necesario.

Ejemplo: Convertir $11111101{,}100010_2$ a octal

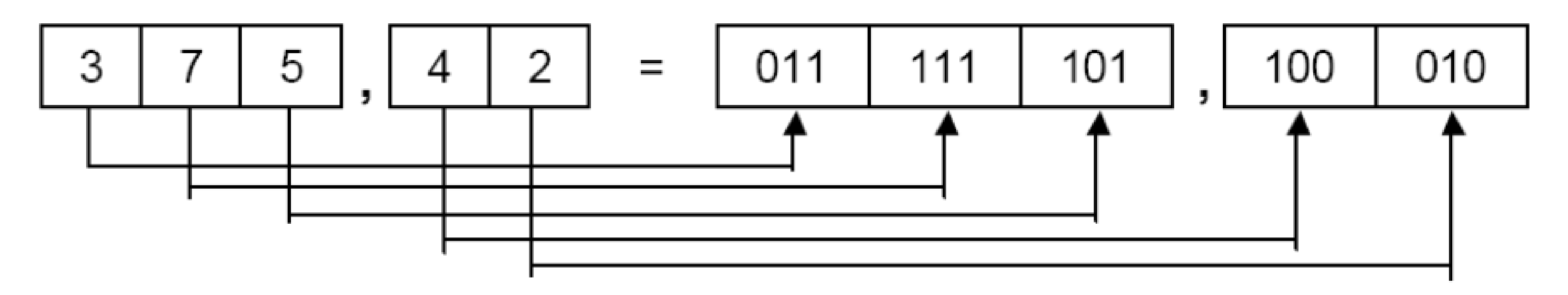


Con lo que tenemos que $11111101{,}100010_2 = 375{,}42_8$

Octal - Binario

Para convertir un numero expresado en base 8 a base 2, simplemente sustituimos cada una de las cifras que lo forman por sus tres cifras binarias equivalentes.

Ejemplo: Convertir a Binario el número $375,42_8$



Con lo que tenemos que $375,42_8 = 011111101,100010_2$

Binario (La Suma)

Sencillas reglas de la suma binaria de un bit.

1+1 = 0 y llevo 1

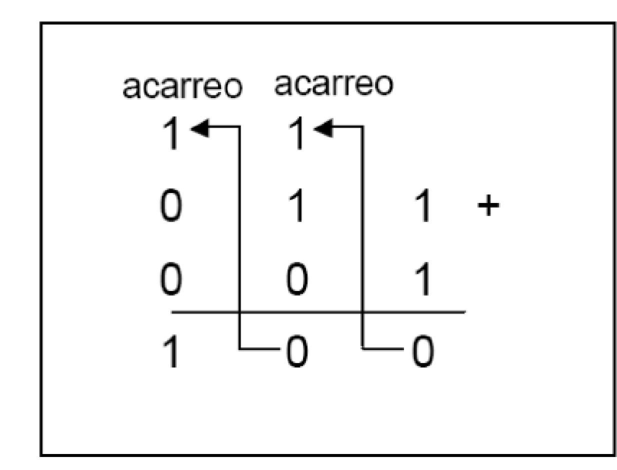
1+0 = 1

0+1 = 1

0+0 = 0

1+1+1 = 1 y llevo 1

Ejemplo de Suma Binaria

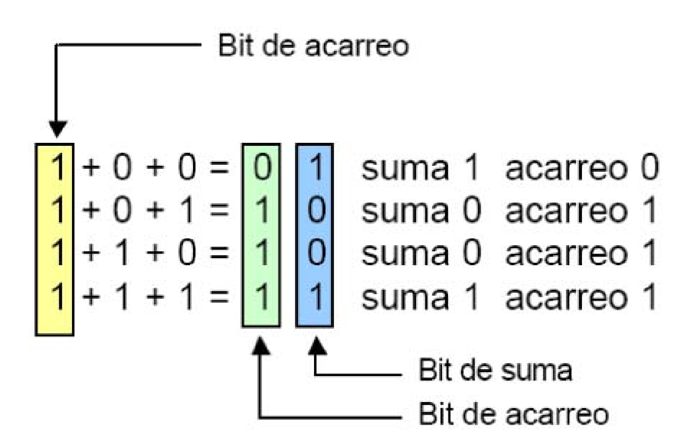


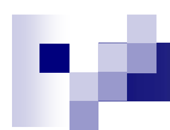
En la columna de la derecha 1 + 1 = 0 con acarreo 1, que pasa a la siguiente columna de la izquierda.

En la columna central, 1 + 1 + 0 = 0 con acarreo 1 que pasa a la siguiente columna de la izquierda. Y en la columna de la izquierda 1 + 0 + 0 = 1.

Ejemplo de Suma Binaria

Cuando existe un acarreo igual a 1 se produce una situación en la que hay que sumar tres bits, un bit correspondiente a cada uno de los números y un bit de acarreo de esta forma tenemos:





Ejemplos de Sumas Binarias

```
C           101111000
x    190     10111110
Y  + 141   + 10001101
------------------------
+y   331    101001011
```

```
c             001011000
X     173      10101101
Y   +  44    + 00101100
------------------------
X +Y  217      11011001
```

```
c           011111110
X    127     01111111
y   + 63    + 0011111
------------------------
+Y   190     10111110
```

```
c             000000000
X     170      10101010
Y    + 85    + 01010101
------------------------
X +Y  255      11111111
```

Binario (La Resta)

Sencillas reglas de la resta binaria de un bit.

0 - 0 = 0

1 - 1 = 0

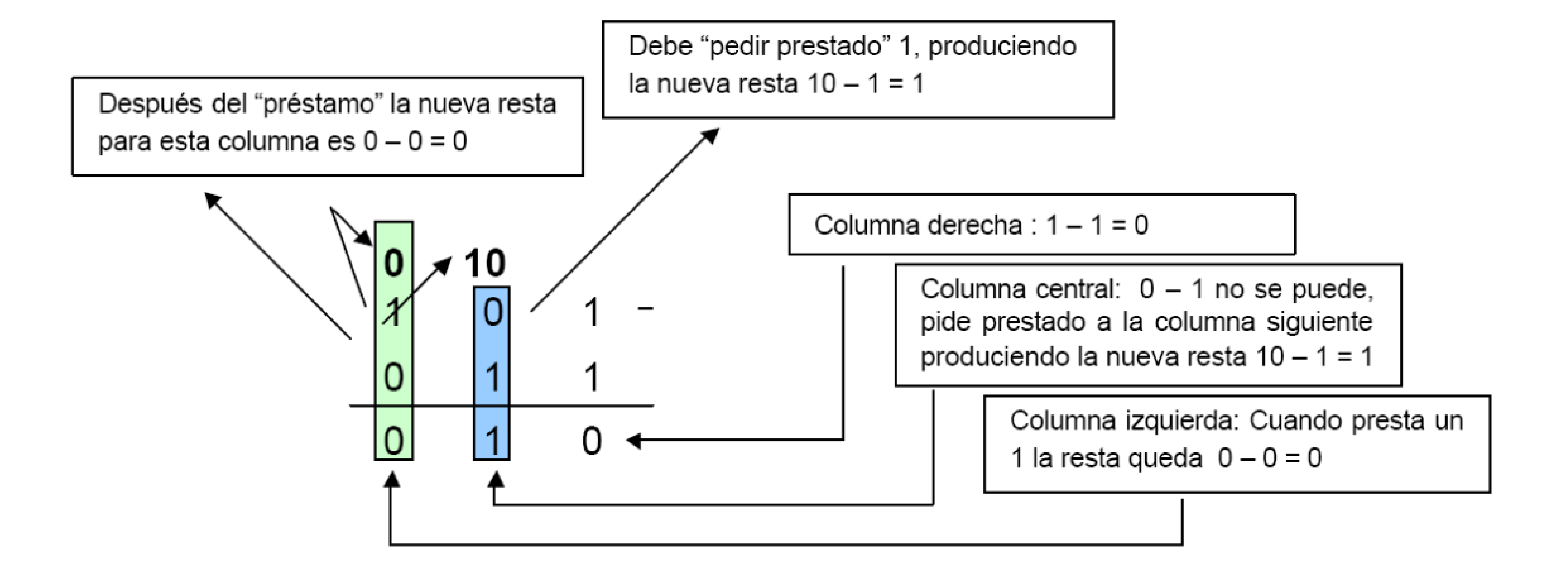
1 - 0 = 1

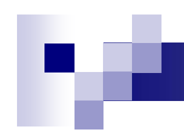
(1)0 - 1 = 1 con acarreo negativo (préstamo) de 1

Ejemplo de Resta Binaria

Cuando se restan números, algunas veces se genera un acarreo negativo que pasa a la siguiente columna de la izquierda. En binario esto sucede cuando se intenta restar 1 de 0. En este caso se pide prestado un 1 de la siguiente columna de la izquierda, y en la columna que se está restando se genera un 10. Veamos esto con un ejemplo:

Supongamos que queremos realizar la resta 5 – 3 = 2 en binario. Esto es 101 – 011

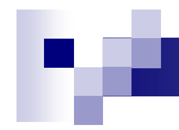




Ejemplos de Restas Binarias

19	10011	63	111111
-11	- 1011	- 46	- 101110
08	01000	17	010001

437	110110101	70	1000110
- 340	- 101010100	93	1011101
097	001100001		



Binario (La Multiplicación)

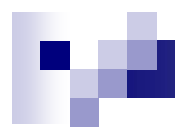
La Multiplicación de Números Binarios se realiza de la misma forma que con números decimales. Se realizan los productos parciales, desplazando cada producto parcial una posición a la izquierda, y luego se suman dichos productos. Es aun mas sencilla que con números decimales, ya que en binario tan solo tenemos dos dígitos: 0 y 1. Cuando multiplicamos por 0 obtenemos 0, y cuando multiplicamos por 1 obtenemos el mismo numero.

La Multiplicación Binaria (Ejemplos)

```
      1 0 1 1  ———————  1 1
    * 1 1 0 1  ———————  1 3
    ---------
      1 0 1 1
    0 0 0 0
  1 0 1 1
+ 1 0 1 1
---------------
1 0 0 0 1 1 1 1  ———————  1 4 3
```

```
      1 0 0 1  ———————    9
    * 1 1 0 1  ———————  1 3
    ---------
      1 0 0 1
    0 0 0 0
  1 0 0 1
+ 1 0 0 1
---------------
  1 1 1 0 1 0 1  ———————  1 1 7
```

```
       1 1 0 0
       1 0 1 1  X
    -----------
       1 1 0 0
     1 1 0 0
   0 0 0 0
 1 1 0 0
-----------
1 0 0 0 0 1 0 0
```



Binario (La División)

- Una división puede ser realizada a través del método de substracciones sucesivas, hasta obtener una diferencia igual a cero (en el caso de una división exacta), o un numero menor al divisor.

Binario (Ejemplos con División)

DIVISION BINARIA

Se realiza del mismo modo que la división decimal. Por ejemplo dividir 110 entre 11

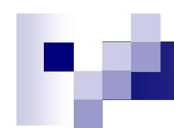
```
 110 | 11
      -----
 11    10
----
  00
  00
```

El resultado será cociente 10 con resto 0

El cociente 10 se forma de la siguiente manera:

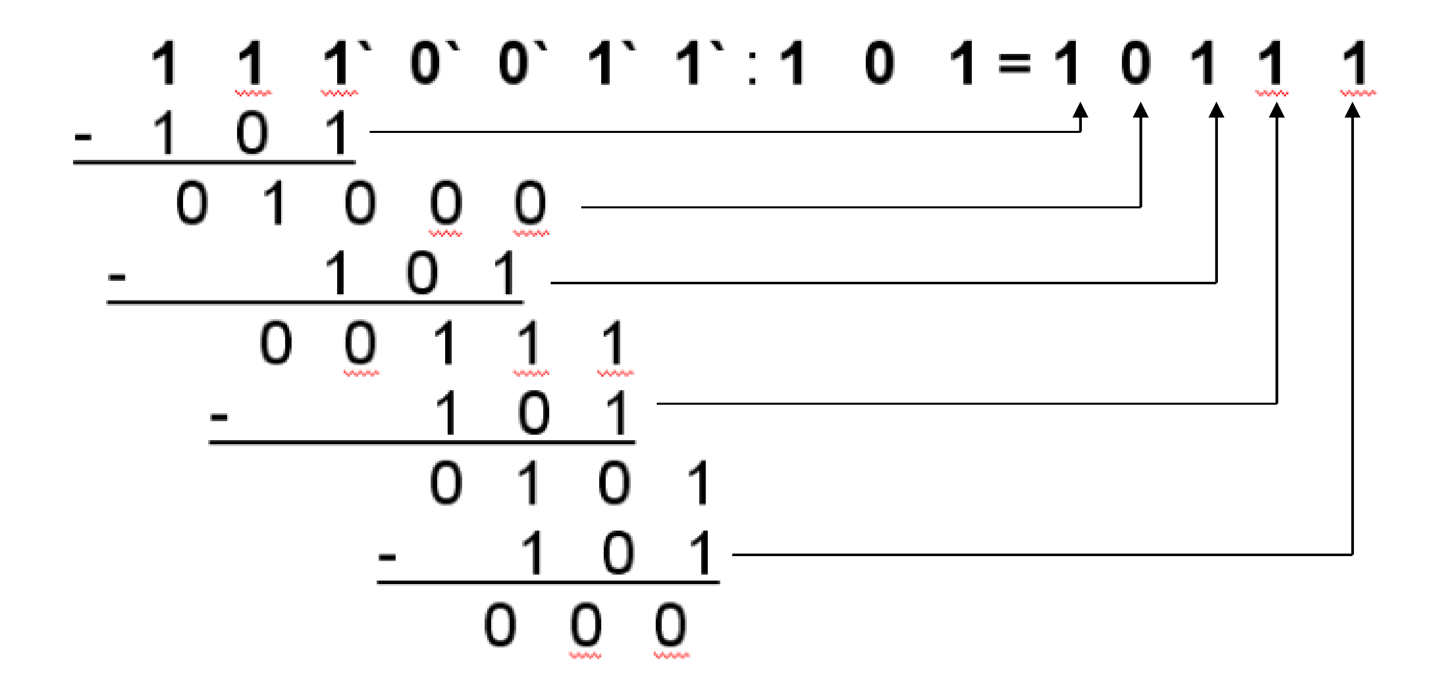
El 1 es resultado de la primera resta (110 -11) se resta una vez y mi cociente se forma con 1

Al bajar el siguiente cero ya no hay posibilidad de volver a restar 11 así que eso da por resultado el siguiente cero hacia la derecha.



Binario (Ejemplos con División)

Dividir 1110011 entre 101

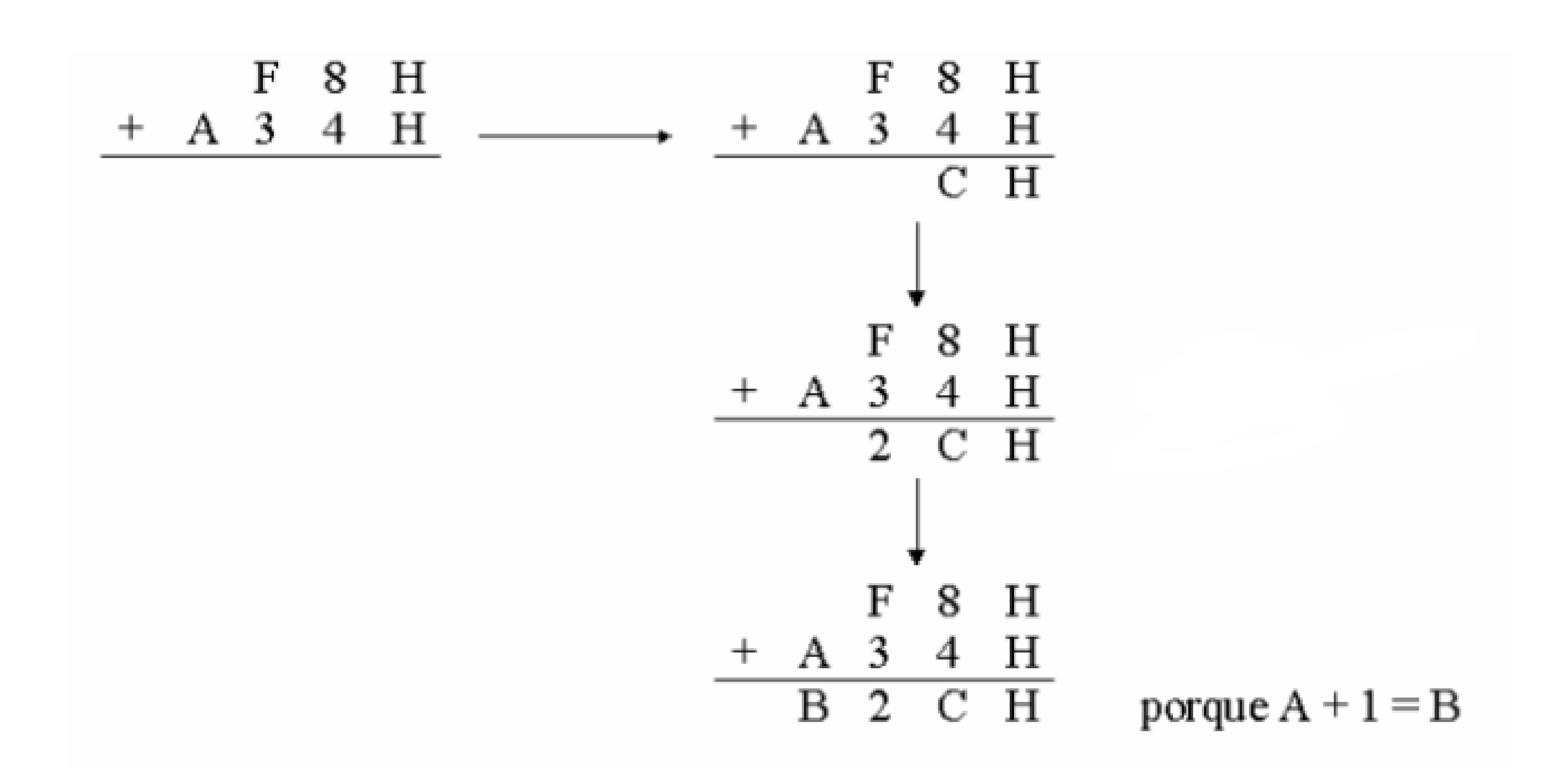


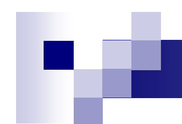
Hexadecimal (La Suma)

+	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10
2	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11
3	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12
4	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13
5	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14
6	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15
7	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16
8	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17
9	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18
A	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19
B	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A
C	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B
D	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C
E	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D
F	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E

Considerar la siguiente tabla para realizar la suma Hexadecimal

Hexadecimal (Ejemplo de Suma)





Hexadecimal (La Resta)

Resta en hexa

	0	16		B	11
	1	6	B	C	1
-		A	1	0	F
	0	C	A	B	2

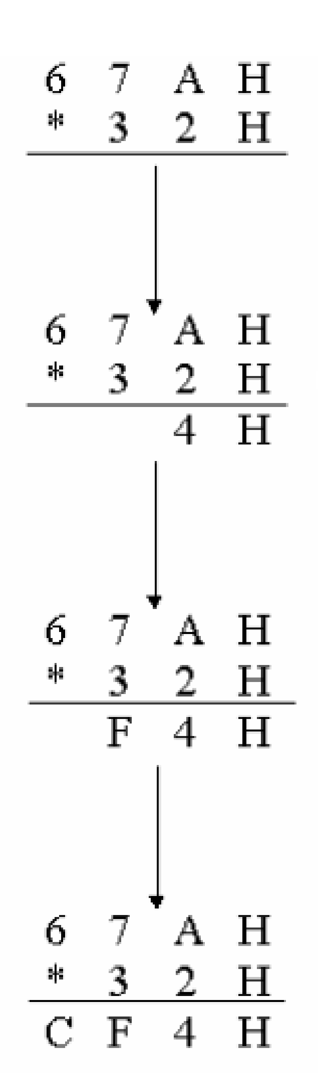
Decimal	Hexadecimal
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	A
11	B
12	C
13	D
14	E
15	F

Hexadecimal (La Multiplicación)

La Multiplicación Hexadecimal se realiza de la misma forma que la multiplicación decimal, utilizando la siguiente tabla:

	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
2	02	04	06	08	0A	0C	0E	10	12	14	16	18	1A	1C	1E
3	03	06	09	0C	0F	12	15	18	1B	1E	21	24	27	2A	2D
4	04	08	0C	10	14	18	1C	20	24	28	2C	30	34	38	3C
5	05	0A	0F	14	19	1E	23	28	2D	32	37	3C	41	46	4B
6	06	0C	12	18	1E	24	2A	30	36	3C	42	48	4E	54	5A
7	07	0E	15	1C	23	2A	31	38	3F	46	4D	54	5B	62	69
8	08	10	18	20	28	30	38	40	48	50	58	60	68	70	78
9	09	12	1B	24	2D	36	3F	48	51	5A	63	6C	75	7E	87
A	0A	14	1E	28	32	3C	46	50	5A	64	6E	78	82	8C	96
B	0B	16	21	2C	37	42	4D	58	63	6E	79	84	8F	9A	A5
C	0C	18	24	30	3C	48	54	60	6C	78	84	90	9C	A8	B4
D	0D	1A	27	34	41	4E	5B	68	75	82	8F	9C	A9	B6	C3
E	0E	1C	2A	38	4C	54	62	70	7E	8C	9A	A8	B6	C4	D2
F	0F	1E	2D	3C	4B	5A	69	78	87	96	A5	B4	C3	D2	E1

Hexadecimal (Ejemplo de Multiplicación)

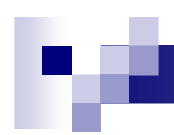


En la tabla vemos que A*2 = 14, entonces escribimos "4" y llevamos 1

2H *7H = 0EH, 0EH + 1 (que accarreo) = 0FH escribo "F" y no llevo o accarreo nada.

2H * 6H = 0CH, se escribe "C" y no hay accarreo.

Continúa



Hexadecimal (Ejemplo de Multiplicación)

Continuación

		6	7	A	H
		*	3	2	H
		C	F	4	
1	3	6	E		

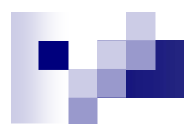
Se continua multiplicando ahora por tres, para obtener el siguiente producto parcial.

		6	7	A	H
		*	3	2	H
		C	F	4	
1	3	6	E		
1	4	3	D	4	H

Una vez obtenidos los dos productos parciales, se procede a realizar la suma.

Hexadecimal (Ejemplo de Multiplicación)

			B	1	2	H
		*	3	F	C	H
		8	4	D	8	
	A	6	0	E		
2	1	3	6			
2	C	1	B	B	8	H



Hexadecimal (La División)

2	F	1	2	
	B	2		2 * 2 = 4; F - 4 = B
2	F	1	2	
0	B	2		2 * 1 = 2; 2 - 2 = 0

Hexadecimal (La División)

```
3  F  4   | 1  A
   B      |------
            2
```

2 * A = 4 (y llevamos uno; F -4 = B

```
3  F  4   | 1  2
0  B      |------
            2
```

2 * 1 = 2 + 1 (de acarreo) = 3; 3 – 3 = 0

```
3  F  4   | 1  A
0  B  4   |------
   1  8     2  6
```

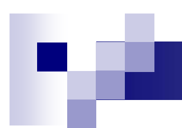
6 * A = C (llevamos tres); 4 – C = 8 (llevamos 4)

6 * 1 = 6 + 4 (de acarreo) = 10; B – A = 1

Hexadecimal (Ejemplo de División)

	5 9 . F …
1 5 4	7 7 8 6 . 0 0
	6 A 4
	0 D 4 6
	B F 4
	1 5 2 0
	1 3 E C
	0 1 3 4 0 …

	89.9…
2125	191237.5
	-17000
	21237
	-19125
	02112 5
	-1912 5
	0200 0



Octal (La suma)

		1					← Acarreo
	3	5	7	0	.	3	1912.375
+			2	5	.	2	+ 21.250
	3	6	1	5	.	5	1933.625

Octal (La Resta)

				10			← Préstamo
	3	5	7	0	.	3	
-			2	5	.	2	
	3	5	4	3	.	1	

	1912.375
-	21.250
	1891.125

Octal (La Multiplicación)

2 * 0 = 1	3 * 0 = 0	4 * 0 = 0	5 * 0 = 0	6 * 0 = 0	7 * 0 = 0
2 * 1 = 2	3 * 1 = 3	4 * 1 = 4	5 * 1 = 5	6 * 1 = 6	7 * 1 = 7
2 * 2 = 4	3 * 2 = 6	4 * 2 = 10	5 * 2 = 12	6 * 2 = 14	7 * 2 = 16
2 * 3 = 6	3 * 3 = 11	4 * 3 = 14	5 * 3 = 17	6 * 3 = 22	7 * 3 = 25
2 * 4 = 10	3 * 4 = 14	4 * 4 = 20	5 * 4 = 24	6 * 4 = 30	7 * 4 = 34
2 * 5 = 12	3 * 5 = 17	4 * 5 = 24	5 * 5 = 31	6 * 5 = 36	7 * 5 = 43
2 * 6 = 16	3 * 6 = 22	4 * 6 = 30	5 * 6 = 36	6 * 6 = 44	7 * 6 = 52
2 * 7 = 16	3 * 7 = 25	4 * 7 = 34	5 * 7 = 43	6 * 7 = 52	7 * 7 = 61



Octal (La Multiplicación)

		3	5	7	0	.	3
	*			2	5	.	2
			7	3	6	0	6
	2	2	5	3	1	7	
	7	3	6	0	6		
1	1	7	2	7	5	.7	6

	1912.375
*	21.250
	95 61875
	382 4750
	1912 375
	38247 50
	40637.96875

Octal (La División)

```
        3
     ________
756 | 33 010

     3 67

      3670
```

Se procede a buscar cuantas veces cabe 3301 en 756 (calculando que sea tres tenemos…)

3 * 6 = 22; 31 – 22 = 7 (accarreo 3)

3 * 5 = 17(acarreo 3); 30 – 22 = 6 (accarreo 3)

3 * 7 = 25(acarreo 3); 33 – 30 = 3

Continuando la división tenemos ahora que proceder a encontrar cuantas veces cabe 3670 en 756 (calculando que sea 4)

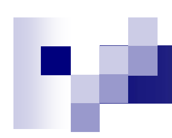
Octal (La División)

```
          3 4
756 | 33 010
      3 67
      3670
       000
```

4 * 6 = 30; 30 – 30 = 0 (acarreo 3)

4 * 5 = 24 (acarreo 3); 27 – 27 = 0 (acarreo 2)

4 * 7 = 34 (acarreo 2); 36 – 36 = 0



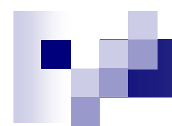
Bibliografía Utilizada

- Introducción a la ciencia de la Computación de la manipulación de datos a la teoría de la computación.
  Behrouz A. Forouzan
  Ed. Thomson
- Arquitectura del Computador
  Prof. Sandro Costantini
  http://medusa.unimet.edu.ve/sistemas/bpis03/clases.htm
- Sistemas Numéricos y representación Interna
  Prof. Roberto Willrich
  http://www.inf.ufsc.br/~willrich/

Lecturas y Actividades Recomendadas

- Lectura de Capítulos 2 y 3 de Introducción a la ciencia de la Computación de Behrouz Forouzan.(págs 14 – 49)

- Resolución de ejercicios, preguntas de opción múltiple y preguntas de repaso de los capítulo arriba mencionados (págs 25 – 26 y págs 45 – 40)



Actividades de la Unidad

- Conversiones TAREA1
- Operaciones Binarias TAREA 2
- Operaciones Hexadecimales TAREA3
- Operaciones Octales TAREA 4